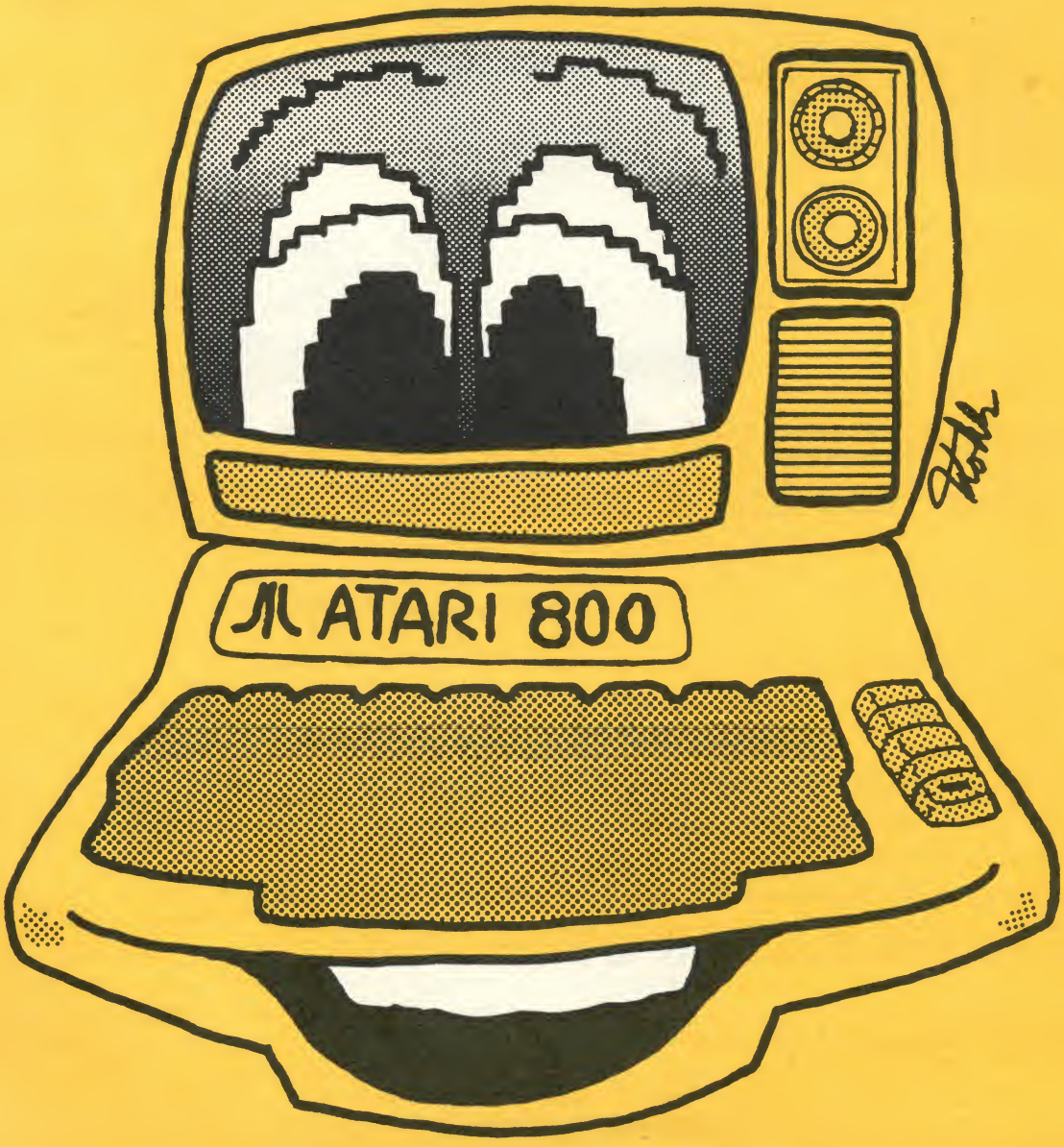
November, 1982
NEWSLETTER

$2.00
Vol. 2, No. 11

MICHIGAN ATARI COMPUTER ENTHUSIASTS

INSIDE:

Talk is Cheap!



The imagination of a child can see further than the sharpest eye

But to see into the future she'll need more than imagination; she'll need skills and knowledge

at
The Family Computer Center
We Offer:

COMPUTER SUPPORT SERVICES℠

- Free Introductory Training
- Hotline Service
- Reduced Cost Extended Warranty
- Technical Library
- Trade Up Assistance

Pay our everyday low price complete with **Computer Support Services℠**

or

Present the lowest advertised price* for any computer listed below and we'll match it when you pay $25 for **Computer Support Services℠**

ATARI 400, ATARI 800, COMMODORE VIC 20, TIMEX 1000, TEXAS INSTRUMENTS TI99/4A

* Must show local newspaper advertisement dated 7 days prior to be eligible. Offer Good Thru October 31, 1982.

Turn Your Child's Imagination Into Creative Reality

THE FAMILY COMPUTER CENTER

at

The Doll Hospital & Toy Soldier Shop

3895 W. 12 Mile Rd., Berkley, 4 Blocks E. of Greenfield **546-8114** **Hours: Mon.-Fri. 10 a.m.-8 p.m., Sat. 10 a.m.-5 p.m., Sun. 1 p.m.-5 p.m.**

INEXPENSIVE VOICE I/O FOR THE ATARI

by

Ed Stewart
11025 Sagebrush Ave
Uniontown Ohio 44685

(Note: This article will appear in a somewhat different form in a future issue of ANTIC magazine. It may not reproduced or reprinted without the expressed permission of both the Author and ANTIC. - The Editor.)

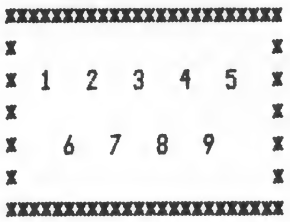
For about ten to fifteen dollars you can make your ATARI computer speak. In fact you can make it whistle, hum, sing or curse. You can save the speech on a disk for posterity and play it back anytime you like. Sound too good to be true? Well, there are some limitations to this technique but for the most part it works very well. I've been playing around with it for some time and decided to share it with other ATARI fans.

Let me start by telling you that I am no hardware expert. I had been looking for an A/D device for the ATARI and discovered that the ATARI has a built in A/D converter. In fact it has 8 of these devices - the ports for the paddle controllers. This led me to an article in the April 1980 issue of BYTE titled "Apple Audio Processing" by Mark Cross. In this article MR. Cross described a circuit he used to input audio data to an Apple computer. The four game paddle inputs on the APPLE are similar to the ATARI paddle ports. They are both A/D converters which generate a count in response to a circuit resistance. The ATARI paddle ports are actually potentiometers (variable resistors) whose resistance changes as the paddle is rotated. If this resistance is changed fast enough and if it has a direct relationship to the sound waves produced by speech then we can read the "paddle" values and store the numbers related to the input speech patterns.

The simple curcuit presented by Mr. Cross converts the voltage produced by a microphone into a variable resistance which is read by the computer. This number is then saved and sent to the TV speaker to produce a voltage whic is in direct proportion to the paddle resistance. The result is recognizable reproduction of whatever sounds are uttered through the microphone.

ATARI HARDWARE

A 9-pin female connector is needed to plug into the joystick/paddle port. APX-9001 is a good plug and it sells for $6.25. The software presented here requires that this plug go into the 3rd port from the left. A description of the controller port pins can be found in section III of the ATARI Hardware Manual. Following is a brief description of the console end (male):



PIN NUMBER	
1-4	joystick input
5	POT B
6	trigger
7	+5 Volts
8	Ground
9	POT A

A double stranded insulated wire about two to three feet long should be soldered to your female plug with one wire to pin 9 and the other to pin 7 (+5V). The other end of the double wire will enter into the circuit described by Mr Cross.

The POKEY chip is responsible for converting the POT resistances into numbers. The conversion technique used is called a voltage-to-frequency conversion and is most noted for it's inexpensive cost and relative inaccuracy. This inaccuracy is recognized in the associated speaker output as noise and hiss although most of the original sound remains intact. The POKEY normally requires 228 TV scan lines to read the POT. The number which is produced by POKEY is actually a count of the number of scan lines it took to charge a capacitor located in POKEY. If there is little resistance (paddle knob turned right) the capacitor charges in fewer scan lines and the resultant number is smaller. A fast POT scan mode is available which causes the capacitors to charge up in only two scan lines. This fast POT

(continued)

scan is what we use to read the values from the microphone because the values must be read thousands of times per second and the normal POT scan mode is much too slow.

OTHER HARDWARE

The following items are needed to construct the circuit in figure 1:

1 moving coil type microphone found in cassette recorders
1 0.1 uF nonpolarized capacitor
1 NPN transistor 2N2222
1 2M Ohm potentiometer
1 100K fixed resistor

The small AC current generated through the microphone causes the base current to change through the transistor. This in turn changes the paddles effective resistance. The potentiometer is used to "tune" the circuit during program execution. After building this circuit and connecting it to the female port connector you are ready to talk to your computer.

VERTICAL BLANK AND DMA

Every 60th of a second whatever program that is executing in your ATARI computer is interrupted by the operating system to perform miscellaneous housekeeping functions. The nature of converting data through the microphone requires that this interruption not occur. IF the vertical blank process were not disabled the speech would be unacceptably choppy. The ANTIC chip is continuously fetching screen data from RAM and displaying it on the TV screen. This process is called DMA. This DMA process is also not acceptable during operation of the speech because it slows down the CPU considerably. There is also a period of time when the DMA does not occur (during vertical blank) and this also would cause an unpredictably choppy sound. The program sample provided turns both DMA and vertical blank exits off during sound input and output processing.

EXECUTING THE PROGRAM

The sample program will display the following MENU:

1 TALK A BIT
2 PLAY BACK THE BIT
3 TALK A LOT
4 PRINT SUMMARY
5 PRINT THE NUMBERS
6 SAVE THE TALK
7 RESTORE THE TALK

After entering the MENU number the program will ask:

WHAT SAMPLE SPEED?

The sample speed can be from 1-255 and causes slower sampling during creation and slower playback with higher numbers.

1 TALK A BIT

After entering the sample speed press the START key. The screen will black out and you will have about 7 seconds of speech. You will need to adjust the potentiometer to get the best quality speech at this point. With a sample speed of 1 the sampling rate is about 4500 samples per second. Increasing the sample speed number will increase the speech duration but will also decrease the quality of the produced sound. The RAM buffer for the speech is 16K bytes and resides from 16K-32K in your machine. You must therefore have at least 40K RAM to run this pogram with no alterations.

2 PLAY BACK THE BIT

This will play back whatever is in the RAM buffer. Playing back with a sample speed of 1 is at least ten times faster than Chip and Dale.

3 TALK A LOT

This is the same as talk a bit except that you can talk indefinitely and have the speech heard through the TV speaker. Press SYSTEM RESET to get out of this.

4 PRINT SUMMARY

Will cause a count of each of the numbers found in the 16K buffer to be printer.

5 PRINT THE NUMBERS

Will print all 16K worth of digitized speech.

(continued)

6 SAVE A TALK

Will cause the 16K buffer to be saved to the specified file.

7 RESTORE A TALK

Will cause a saved 16K buffer to be restored to RAM. You can then use MENU item 2 to play back the restored speech.

COMPRESSION

The number coming from the POT controller is 8 bits and the number sent to the TV speaker is 4 bits. This program has therefore packed two input voltages into a single byte in the RAM buffer. Other compression techniques such as compression of like characters and Hamming Codes would have to be implemented if this method of generating speech were to be fully implemented.

ATARI D/A CONVERTER

The ATARI computer has built into it a 4 bit D/A converter for each of the four audio output voices. By setting the proper switch this program has used audio channel number one in this mode. This in no way affects what is happening in the other three voices. Setting address $D201 to a $1X causes the X to be output to the TV speaker as a voltage. X can be a number from zero to fifteen. Each different X sent to this address causes the TV speaker cone to be moved in or out. If these X's are sent fast enough to $D201 then the speaker cone moves very fast and we have sound at frequencies we can hear. This is exactly what our program does with it's digitized input.

CONCLUSION

This technique is quite inexpensive and gives one a taste of the machine and it's capabilities and limitations. It is fairly easy to implement and the results are reasonably satisfactory. However if excellent speech is desired it is necessary to obtain a high quality A/D converter for accurate digitization. To the best of my knowledge there is no commercially available A/D for the ATARI computer yet. When this item becomes available you will have a head start with the experience you may have gained using the technique outlined here and you may more thoroughly understand the workings of your ATARI computer.

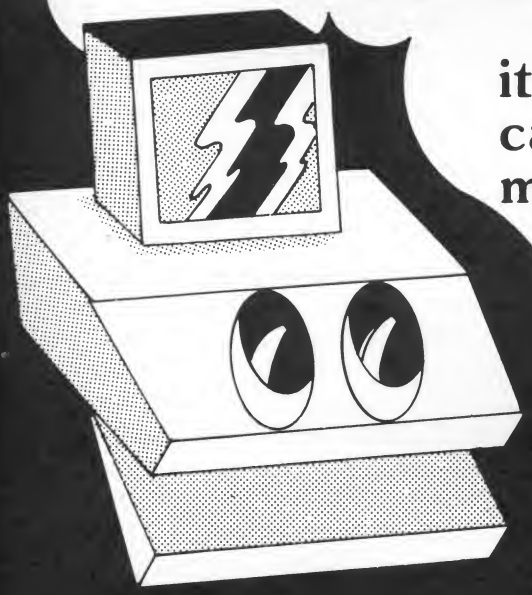
(continued)

ATARI VOICE I/O
ASSEMBLER SOURCE

```
10 .OPT   LIST,NOEJECT
20 *= $600
30 PLA PULL OFF DUMMY ARG COUNT
40 LDA #$08
50 STA $D01F INIT SWITCHES
60 M1 LDA $D01F GET VALUE
70  AND #$01 ?START PRESSED
80  BNE M1 NOT YES
90  LDY #$FF DELAY
0100 M2 LDX #$FF SOME
0110  JSR DEL GO LOOP
0120  DEY
0130  BNE M2
0140  LDA #$08  RESET
0150  STA $D01F START SWITCH
0160  LDX 208   GET PARM
0170  CPX #0 ?PLAYBACK
0180  BNE NP NO
0190  JMP PB GO PLAY BACK SOUND
0200 NP LDA #0
0210  STA $D400 KILL DMA
0220  STA $D40E KILL VBI
0230 MD STA $D40A WSYNC
0240  STA $D40A WSYNC
0250 DONE LDX 207
0260  JSR DEL GO WAIT IF NEED BE
0270  LDA $D204 GET INPUT BYTE
0280 GO LDX #$13
0290  STX $D20F TURN OFF FAST SCAN
0300  LDX #$17
0310  STX $D40A WSYNC
0320  STX $D20F SAY FAST POT SCAN
0330  STX $D20B START SCAN
0340  LDX FLAG
0350  CPX #0 ?LEFT HALF OF BYTE
0360  BNE RT NO
0370  AND #$F0
0380  STA BYTE SAVE IT
0390  ROR A
0400  ROR A
0410  ROR A
0420  ROR A
0430  AND #$0F
0440  ORA #$10 SAY USE 4 BIT D/A
0450  STA $D201 ECHO TO SPEAKER
0460  INC FLAG SAY RIGHT NEXT
0470  JMP MD
0480 RT ROR A
0490  ROR A
0500  ROR A
0510  ROR A
0520  AND #$0F
0530  ORA #$10 SAY USE 4 BIT D/A
0540  STA $D201 SAY SOMETHING
0550  AND #$0F REMOVE TOP 4 BITS
0560  ORA BYTE ADD IN LEFT NIBBLE
0570  DEC FLAG SAY LEFT NEXT
0580  LDY #0
0590  STA (205),Y SAVE TO BYTE
0600  LDA $D01F
0610  AND #$01 ?START KEY
0620  BEQ FINI YES
0630 D3 INC 205 INCREMENT
0640  BNE MD BUFFER
0650  INC 206 AREA
0660  LDX 206 POINTER
0670  CPX 209 ?END OF MEMORY
0680  BNE MD  NO CONTINUE
0690  JMP FINI GO FINISH THINGS
0700 DEL DEX
0710  BNE DEL
0720  RTS
0730 FINI LDA 208 ?TALK A LOT
0740  CMP #2 ?HUH
0750  BNE FINI1 NO, JUST RETURN
0760  LDA #0 RESET
0770  STA 205 BUFFER
0780  LDA #64 START
0790  STA 206 POINTERS
0800  JMP NP AND DO IT AGAIN
0810 FINI1 LDA #$40
0820  STA $D40E RESTART VBI'S
0830  LDA #$22 AND SCREEN DMA
0840  STA $D400
0850  RTS
0860 PB LDA #0
0870  STA $D40E KILL VBI
0880  STA $D400 KILL DMA
0890 PB1 LDX 207 GET DELAY COUNT
0900  JSR DEL
0910  LDY #0
0920  LDA (203),Y GET SOME DATA
0930  TAX
0940  ROR A
0950  ROR A
0960  ROR A
0970  ROR A
0980  AND #$0F GET LEFT NIBBLE
0990  ORA #$10 SAY USE D/A
1000  STA $D201 STORE THE VALUE
1010  TXA
1020  AND #$0F
1030  ORA #$10
1040  CLC
1050  CLC
1060  CLC
1070  CLC
1080  LDX 207 GET DELAY VALUE
1090  JSR DEL
1100  STA $D201 MAKE SOME NOISE
```

(continued)

yes, virginia... there is a store that sells software



it is
called
micro station

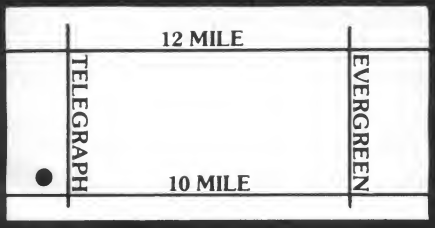
we have the largest
selection of computer books,
magazines, supplies & software around

accounting • business • data base
education • entertainment • home/hobby
technical • word processing

make it easy for people to select the things you really want
sign up at our gift registry

gift certificates available

open after the meeting for your shopping pleasure



PRESENT YOUR MEMBERSHIP CARD FOR SPECIAL PRICE

Mon-Fri 11:00-7:00 Sat. 12:00-6:00

Visit our Retail Store

24484 W. Ten Mile Rd.
(½ blk. W. of Telegraph)
Southfield, MI 48034
(313) 358-5820

micro station

```
1110 D4 INC 203 INCREMENT
1120  BNE PB
1130  INC 204 BUFFER
1140  LDX 204
1150  CPX 206 POINTERA
1160  BNE PB1
1170  JMP FINI
1180 BYTE .BYTE 0
1190 FLAG .BYTE 0
```

VOICE I/O PROGRAM

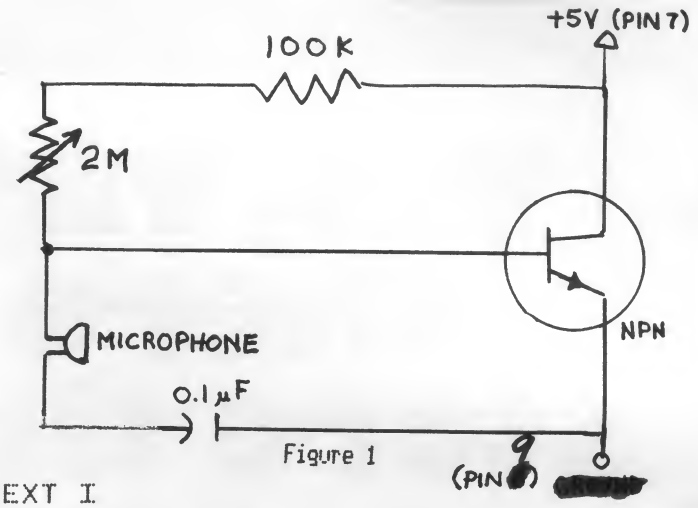


Figure 1

```
20 FOR I=0 TO 243:READ Z:POKE 1536+I,Z:NEXT I
40 GOTO 140
60 FOR I=16384 TO 32767:Z(PEEK(I))=Z(PEEK(I))+1:NEXT I:POKE 54272,34:POKE 559,34
80 ? #3;CHR$(27);CHR$(56);CHR$(29);
100 FOR I=0 TO 255:? #3;I;"-";Z(I);"    ";
120 NEXT I:CLOSE #3:GOTO 160
140 DIM Z(255),FN$(13)
160 GRAPHICS 0:? "OPTION--> 1 TALK A BIT":? "           2 PLAY BACK THE BIT":? "
       3 TALK A LOT"
180 ? "           4 PRINT SUMMARY":? "           5 PRINT THE NUMBERS"
200 ? "           6 SAVE A TALK":? "          7 RESTORE A TALK"
220 TRAP 220:INPUT ANS:IF ANS>7 THEN 220
240 TRAP 240:? "WHAT SAMPLE SPEED":INPUT SS:IF SS>255 THEN 240
260 ON ANS GOTO 280,360,460,340,420,500,580
280 POKE 208,1:POKE 205,0:POKE 206,64:POKE 207,SS:POKE 209,128
300 A=USR(1536):POKE 562,3:POKE 53775,3
320 GOTO 160
340 FOR I=0 TO 255:Z(I)=0:NEXT I:POKE 54272,0:OPEN #3,8,0,"P:":POKE 559,0:GOTO 6
0
360 POKE 207,SS:POKE 203,0:POKE 204,64:POKE 208,0:POKE 206,128
380 A=USR(1536):POKE 562,3:POKE 53775,3
400 GOTO 160
420 OPEN #3,8,0,"P:":? #3;CHR$(27);CHR$(56);CHR$(29);:FOR I=16384 TO 32767:? #3;
PEEK(I);" ";:NEXT I
440 CLOSE #3:GOTO 160
460 POKE 208,2:POKE 205,0:POKE 206,64:POKE 207,SS:POKE 209,128
480 A=USR(1536):GOTO 460
500 ? "GIVE FILE NAME";:INPUT FN$
520 TRAP 500:OPEN #4,8,0,FN$:TRAP 560
540 POKE 559,0:FOR I=16384 TO 32767:PUT #4,PEEK(I):NEXT I
560 CLOSE #4:POKE 559,34:GOTO 160
580 ? "GIVE FILE NAME";:INPUT FN$
600 TRAP 580:OPEN #4,4,0,FN$:TRAP 640
620 POKE 559,0:FOR I=16384 TO 32767:GET #4,Z:POKE I,Z:NEXT I
640 CLOSE #4:POKE 559,34:GOTO 160
660 DATA 104,169,8,141,31,208,173,31,208,41,1,208,249,160,255,162,255,32,149,6
680 DATA 136,208,248,169,8,141,31,208,166,208,224,0,208,3,76,181,6,169,0,141
700 DATA 0,212,141,14,212,141,10,212,141,10,212,166,207,32,149,6,173,4,210,162
720 DATA 19,142,15,210,162,23,142,10,212,142,15,210,142,11,210,174,243,6,224,0
740 DATA 208,22,41,240,141,242,6,106,106,106,106,41,15,9,16,141,1,210,238,243
760 DATA 6,76,45,6,106,106,106,106,41,15,9,16,141,1,210,41,15,13,242,6
780 DATA 206,243,6,160,0,145,205,173,31,208,41,1,240,19,230,205,208,163,230,206
800 DATA 166,206,228,209,208,155,76,153,6,202,208,253,96,165,208,201,2,208,11,16
9
820 DATA 0,133,205,169,64,133,206,76,37,6,169,64,141,14,212,169,34,141,0,212
840 DATA 96,169,0,141,14,212,141,0,212,166,207,32,149,6,160,0,177,203,170,106
860 DATA 106,106,106,41,15,9,16,141,1,210,138,41,15,9,16,24,24,24,24,166
880 DATA 207,32,149,6,141,1,210,230,203,208,206,230,204,166,204,228,206,208,206,
76
900 DATA 153,6,0,0
```

BASIC ROUTINES

by Jerry White

Review
By RICHARD GIZYNSKI

Tired of trying to find good examples for program routines? Had enough manuals for awhile? Basic Routines from Adventure International may be the answer for you.

For $24.95 (disk or cassette) you get a package of programs that demonstrate a variety of useful routines. A descriptive manual comes with it but you probably won't need to read past page one. But don't throw it away. The back of the book has a lot of useful programing tips and referance peek/poke positions.

The routines were designed for those who have gotten past their Atari Basic self-teaching guide but aren't ready to design their own arcade game. Just load the routine, press the break key and find out how the routines work by following the REM statements. If you're not up to learning right then, you can play with the routines.

The newest of programmers will find 'ENGLISH' a good example of programing technique. It demonstrates how to set up a menu, how to use subroutines, using understandable variables, making your keyboard warn you of errors and a general style of programming. Lots of REM statements help you find your way around the program.

'PADDLE' and 'JOYSTICK' are demos for...paddle and joystick inputs. They also show two potential game uses for the imputs.

Each program takes you into different aspects of useful routines. The main menu (disk version) has programs uses programs named for what they demo. You can copy the subroutines to a program you are writing and use things like -- a 'TIMER' to clock game responses, 'KEYDEMO' to read the keyboard, 'SORTDEMO' for alphabetizing information, etc. 'TABDEMO' is a handy way for setting up your screen to print out game info and you may want 'RJUSTIFY' amounts in your new checkbook routine.

As you gain experience, you will start working with more complex graphics and sounds. 'MODE123' shows you how to mix graphics modes by modifying the display list. 'GR8TEXT' shows you how to print text to the graphics window in Graphics 8.

Several sound programs provide you with examples of how to put a theme song in your program or how to create the bangs, booms, crashes and special effects sounds you need for that saucer landing.

Not all Atari programs are games, so Jerry White, the John Lee Hooker of Atari programming, and author of this package, put in a demonstration of a random access disk data base. Whew! Quite a mouthful but really useful if you want to learn how to write and manage an inventory system or data base of your own. (A data base is just information stored in a predefined pattern -- like checkbook entries).

The manual is an easy to read overview of what each program is supposed to do. For a few programs, it explains in a little more detail what might be missed by following the REM statements in the program itself. The last six pages are tips on how to speed up your program, save memory and a handy peek and poke guide used in the programs.

If you've bought your Atari within the last year and want to write programs of your own, this collection of routines can be a big help.

YOUR CHOICE $19.99 each LIST $29.95

APPLE MECHANIC	1	LOST COLONY	3
APPLE PANIC	1,2,3	MOON BASE 10	2
BEER RUN	1	PACIFIC COAST HWY	2
BUG ATTACK	1,2	PIG PEN	1,3
CALL TO ARMS	3	PREPPIE	2
CANYON CLIMBER	2	RASTER BLASTER	1,2
CLOWNS & BALLOONS	2	SHOOTING ARCADE	2
CROSSFIRE	1,2	SNAKE BYTE	1,2
CRUSH CRUMBLE CHOMP	1,2	SNEAKERS	1,2
CRYPT OF THE UNDEAD	2	SPACE EGGS	1,2
CYCLOD	1,2	TRACK ATTACK	1,2
DISKMANAGER	2	TUMBLE BUGS	2
Dr. GOODCODES CAVRN	2	TWERPS	1
FAST EDDY (K)	2	UTILITY CITY	1
FREE FALL	1		
GHOST ENCOUNTERS	2		
JAWBREAKER	1,2		
LABYRINTHS	1		

1 = APPLE
2 = ATARI
3 = IBM PC
K = CARTRIDGE

ORDER GIFTS EARLY TO AVOID HOLIDAY SHIPPING DELAYS

See our catalog for other programs at discount prices.

POPULAR SOFTWARE

VISICALC $179

Call for other Business Specials

More Computer Fun

EACH ONLY $22.99 LIST 34.95

BANDITS	1,2
CHOPLIFTER	1,2
FROGGER	1,2,3
KNIGHT OF DIAMONDS	1
POOL 1.5	1,2
SERPENTINE	1,2

EACH ONLY $23.99 LIST 34.95

ANDROMEDA (NEW)	2
CANNONBALL BLITZ	1
DAVID'S MIDNIGHT MG	1,2
FACE MAKER	1,3
MY FIRST ALPHABET	2
MOUSKATTACK	1,2
NAUTILUS	2
PICKNICK PARANOIA	2
SARGON II	1
SHAMUS	2
SLIME	2
STORY MACHINE	1,3
SWASHBUCKLER	1
ULYSSES & GOLDEN FL	1,2,3
WORM WAR I (K)	2

EACH ONLY $26.99 LIST 39.95

MASTERTYPE	1,2
ULTIMA	1,2
ZORK I	1,2,3
ZORK II	1,2,3

EACH ONLY $27.99 LIST 39.95

STARCROSS	2
TEMPLE OF APSHAI	1,2,3
TIGERS IN THE SNOW	1,2
THRESHOLD	1,2
WAYOUT	1,2
WIZARD OF WOR	2

EACH ONLY $31.99 LIST 44.95

CENTIPEDE	2
STAR RAIDERS	2
PACMAN	2

EACH ONLY $33.99 LIST 49.95

DEADLINE	1,2,3
WIZARDRY	1

EACH ONLY $34.99 LIST 49.95

K-RAZY SHOOTOUT (K)	2
K-RAZY KRITTERS (K)	2
K-RAZY ANTICS (K)	2

SPECIALS are good through 12/31/82

Add $2.00 for shipping. MI residents add 4% tax. Money orders, checks (allow 10 days) accepted. MC/VISA accepted with card # & expir. date. Add $1.50 for C.O.D. Prices subject to change.

M-F 10 AM - 9 PM
Sat 10 AM - 6 PM
313 528-1554

FREE CATALOG

HOLIDAY SPECIALS FROM

The Computer Express

P.O. Box 569 , Troy, MI. 48099 (313) 528-1554

Give a GIFT CERTIFICATE from TCE

```
1 REM ** ACE NEWSLETTER
2 REM **3662 VINE MAPLE
3 REM **   EUGENE, OR
4 REM **     97405
6 REM **    Nov 1982
7 REM ********************
10 REM ********************
20 REM ** VULTURES III  **
30 REM **STAN OCKERS **
40 REM ********************
90 GRAPHICS 18:POSITION 5,3:? #6;"vultures":POSITION 8,5:? #6;"III"
100 PMHI=1547:IMAGE0=1552:IMAGE1=1553:IMAGE2=1554:IMAGE3=1555:HPOS3=1559:VPOS3=1567
110 FLAG0=1568:FLAG1=1569:FLAG2=1570
112 COL0=52:COL1=24:COL2=32:COL3=244:BKCOL=144
120 VPOS0=1564:VPOS1=1565:VPOS2=1566:IMGPT=1584
130 HPOS0=1556:HPOS1=1557:HPOS2=1558:RAMTOP=106:PMBASE=54279:SDMCTL=559:GRACTL=53277
140 PCOLR0=704:PCOLR1=705:PCOLR2=706:PCOLR3=707:POKE 1577,10
150 DIM B$(20),BB$(20),V$(2):V$(1)=CHR$(136):V$(2)=CHR$(138):BB$(1)=CHR$(0):BB$(20)=CHR$(0):BB$(2)=BB$
151 DIM CL$(38):CL$(1)=" ":CL$(38)=" ":CL$(2)=CL$
152 DIM X$(38),X1$(38):X$(1)="X":X$(38)="X":X$(2)=X$:X1$(1)="x":X1$(38)="x":X1$(2)=X1$
153 DIM LTNG$(46):RESTORE 154:FOR I=1 TO 46:READ A:LTNG$(I,I)=CHR$(A):NEXT I
154 DATA 124,29,41,29,30,124,29,30,30,40,29,30,30,40,29,30,124,29,40,29,124,29,41,29,41,29,41,29,30,124,29
155 DATA 30,30,41,29,30,30,41,29,30,30,41,29,30,30,124
156 DIM ERAS$(46):RESTORE 157:FOR I=1 TO 46:READ A:ERAS$(I,I)=CHR$(A):NEXT I
157 DATA 32,29,32,29,30,32,29,30,30,32,29,30,30,32,29,30,32,29,32,29,32,29,32,29,32,29,32,29,30,32,29
158 DATA 30,30,32,29,30,30,32,29,30,30,32,29,30,30,32
160 COUNT=20:PERCH=180:BIRDS=0:DIF=3:POKE 1555,8
200 GOSUB 6000:GOSUB 2000:POKE 764,255:? CHR$(28);"PRESS START TO BEGIN":? "ANY KEY TO PAUSE"
202 ? "Beware the golden vultures !!!":GOSUB 1000
210 A=PEEK(RAMTOP)-16:POKE PMBASE,A:POKE PMHI,A:SOUND 0,0,0,0:A=USR(1536):GRAPHICS 0:GOSUB 5000
220 POKE SDMCTL,62:POKE GRACTL,3:POKE 764,255
230 COUNT=20:PERCH=180:BIRDS=0:DIF=3:POKE 1555,8:GOLD=500:SCORE=0
255 SHIELD=20:FOR J=SHIELD TO 23:POSITION 1,J:? X$;:NEXT J:POSITION 0,0:? " dif  score  high"
260 POKE PCOLR0,243:POKE PCOLR1,243:POKE PCOLR2,243:POKE PCOLR3,40
270 POKE 752,1:FOR I=2 TO 36 STEP 2:POSITION I,1:? V$:NEXT I:BLEFT=18:BLAND=0:POKE 1575,0:HIT=0
275 POSITION 1,SHIELD-1:? CL$
277 POSITION 22,0:? DIF:POSITION 26,0:? SCORE:POSITION 34,0:? HIGH
280 RESTORE 285:FOR I=1564 TO 1571:READ A:POKE I,A:NEXT I:POKE 1559,100:POKE 1783,11
285 DATA 0,0,0,150,0,0,0,0
290 B$=BB$
299 POKE 712,BKCOL:POKE 1574,0:POKE 708,COL0:POKE 709,COL1:POKE 710,COL2:POKE 711,COL3
300 COUNT=COUNT-1:A=PEEK(1575):IF HIT<A THEN HIT=A:POKE 1788,129:SCORE=SCORE+10*DIF:POSITION 26
,0:? SCORE
310 IF PEEK(FLAG0)=0 AND COUNT<0 THEN GOSUB 905:POKE (FLAG0),1:POKE (HPOS0),40+R*8:POKE VPOS0,48
320 IF PEEK(FLAG0)=1 THEN POKE VPOS0,PEEK(VPOS0)+DIF
330 IF PEEK(VPOS0)>=PERCH THEN POKE FLAG0,0:A=INT((PEEK(HPOS0)-48)/4):GOSUB 950:POKE HPOS0,0:POK
E VPOS0,32
340 IF PEEK(FLAG1)=0 AND COUNT<0 THEN GOSUB 905:POKE (FLAG1),1:POKE (HPOS1),40+R*8:POKE VPOS1,48
350 IF PEEK(FLAG1)=1 THEN POKE VPOS1,PEEK(VPOS1)+DIF
360 IF PEEK(VPOS1)>=PERCH THEN POKE FLAG1,0:A=INT((PEEK(HPOS1)-48)/4):GOSUB 950:POKE HPOS1,0:POK
E VPOS1,32
370 IF PEEK(FLAG2)=0 AND COUNT<0 THEN GOSUB 905:POKE (FLAG2),1:POKE (HPOS2),40+R*8:POKE VPOS2,48
380 IF PEEK(FLAG2)=1 THEN POKE VPOS2,PEEK(VPOS2)+DIF
390 IF PEEK(VPOS2)>=PERCH THEN POKE FLAG2,0:A=INT((PEEK(HPOS2)-48)/4):GOSUB 950:POKE HPOS2,0:POK
E VPOS2,32
392 IF BLAND>2 THEN 500
394 IF BLEFT<1 THEN 600
400 FLASH=FLASH-1:IF FLASH<0 THEN GOTO 550
405 IF PEEK(1574)>0 THEN GOTO 650
```

VULTURES III

BY STAN OCKERS

```
410 GOLD=GOLD-1:IF GOLD<0 AND BLAND>3 THEN GOLD=100*(9-DIF):DIF=DIF+2:XCNT=30:POKE 704,44:POKE
 705,44:POKE 706,44
420 IF XCNT>0 THEN XCNT=XCNT-1:IF XCNT=0 THEN DIF=DIF-2:POKE 704,243:POKE 705,243:POKE 706,243:PO
SITION 22,0:? DIF
425 IF PEEK(764)<255 THEN GOSUB 1000
430 GOTO 300
499 REM ** REMOVE A SHIELD **
500 POKE 1789,207:SVCOL=COL1:FOR I=1 TO 6:POKE 708,PEEK(708)+2:POSITION 1,SHIELD:? X1$:FOR J=1 TO 5
:NEXT J
502 POSITION 1,SHIELD:? X$:FOR J=1 TO 5:NEXT J:NEXT I:POSITION 1,SHIELD-1:? CL$:POSITION 1,SHIELD:?
 CL$
508 SHIELD=SHIELD+1:PERCH=PERCH+8:IF SHIELD=23 THEN 700
510 DIF=DIF-1:IF DIF<2 THEN DIF=2
520 GOTO 270
550 POKE 1789,171:POKE 712,40:X=RND(0)*23+8:POSITION X,2:? LTNG$;:POKE 712,15:POSITION X,2:? ERAS$;:P
OKE 712,BKCOL
555 A=9-DIF:FLASH=RND(0)*30*A+10*A
560 GOTO 405
599 REM ** REMAINING BIRDS LEAVE **
600 DIF=DIF+1:IF DIF>6 THEN DIF=6
612 FOR I=1 TO 150
614 P=PEEK(VPOS0):IF P>0 THEN POKE VPOS0,P-1
616 P=PEEK(VPOS1):IF P>0 THEN POKE VPOS1,P-1
618 P=PEEK(VPOS2):IF P>0 THEN POKE VPOS2,P-1
620 NEXT I:POKE 77,0:GOTO 270
649 REM ** HIT BY LIGHTNING **
650 POKE HPOS3,0:POKE 53278,0:POKE 1789,189:POKE 707,BKCOL
655 FOR I=8 TO 48 STEP 4:POKE 712,I:FOR J=1 TO 30:NEXT J:NEXT I:POKE 712,BKCOL
660 FOR I=1 TO 200:NEXT I:POKE 1788,79:FOR J=1 TO DIF
665 SCORE=SCORE-100:IF SCORE<0 THEN SCORE=0
670 POSITION 26,0:? SCORE;"   ":FOR L=1 TO 100:NEXT L:NEXT J
690 POKE 707,40:GOTO 270
700 IF SCORE>HIGH THEN HIGH=SCORE
705 POKE GRACTL,0:FOR I=53261 TO 53264:POKE I,0:NEXT I
710 GRAPHICS 18:POSITION 5,3:? #6;"final score "
720 POSITION 8,5:? #6;SCORE
730 POSITION 5,8:? #6;"PRESS START":POSITION 4,9:? #6;"TO PLAY AGAIN"
740 IF PEEK(53279)<>6 THEN 740
750 GRAPHICS 0:GOSUB 5000:GOTO 220
900 REM ** START ANOTHER BIRD **
905 COUNT=INT(RND(0)*30)
910 R=INT(RND(0)*18)+2:IF ASC(B$(R))>0 THEN 910
920 B$(R,R)=CHR$(1):POSITION 2*(R-1),1:? "  ":BLEFT=BLEFT-1:RETURN
949 REM ** BIRD AT SHIELD **
950 POSITION A,(PERCH-32)/8:? V$;:POKE 1788,150:BLAND=BLAND+1:RETURN
999 REM ** PAUSE ROUTINE **
1000 POKE 764,255
1010 IF PEEK(53279)<>6 THEN 1010
1020 RETURN
1999 REM ** VBI ROUTINE **
2000 RESTORE 2001:DIM VB$(348):FOR I=1 TO 348:READ A:VB$(I,I)=CHR$(A):NEXT I
2001 DATA 72,138,72,152,72
2002 DATA 206,40,6,173,40,6,16,29,173,41,6,141,40,6,162,2,254,16,6,254,16,6,189,16,6,201,8,144,5,169,0,157,16,6
2004 DATA 202,16,235
2005 DATA 174,120,2,224,11,208,3,142,247,6,224,7,208,3,142,247,6
2006 DATA 174,247,6,173,23,6,201,200,176,10,224,7,208,6,238,23,6,238,23,6,201,48,144,10,224,11,208,6
2008 DATA 206,23,6,206,23,6
2010 DATA 24,173,11,6,105,4,133,204,162,0,134,207,160,0,132,203,189,20,6,157,0,208,189,12,6,221,16,6,208,8,189,2
8
```

NOW THE ATARI® 800™ HOME COMPUTER

OPEN
10 a.m.-9 p.m.
Mon.-Sat.

THE DISCOUNT
Computer SOURCE

GIFT CERTIFICATES

PRINTERS

Modems

Monitors

48K

$639.

EVERYTHING DISCOUNTED EVERYDAY

SOFTWARE

EDU-WARE
OPTIMIZED
DATASOFT
AVALON HILL
BRODERBUND
SYNAPSE
VISICORP
QUALITY
DON'T ASK
GEBELLI
ON-LINE
INFOCOM
LJK
IN-HOME
K-BYTE
SIRIUS
STRATEGIC SIMULATIONS
ATARI PROGRAM EXCHANGE
AUTOMATED SIMULATIONS
PROGRAM DESIGN INC.
AND MANY MORE

NOW THRU CHRISTMAS
OPEN SUNDAY
12 TO 5

HIGH PERFORMANCE FLOPPY DISK $1.79

ATARI® ACCESSORIES

ATARI 800 48K $639.00
ATARI 400 16K $269.00
ATARI 410 REC. $ 85.
ATARI 810 DISK $449.
ATARI 830 MODEM $149.
ATARI 850 INTERFACE $179.

OVER 500 PROGRAMS IN STOCK

RITE WAY ENTERPRISES

8262-12 MILE RD.
WARREN MI. 48093
313-751-2454

22027 MICHIGAN AVE.
DEARBORN MI. 48124
313-562-3178

```
2020 DATA 6,221,24,6,240,69,189,16,6,157,12,6,189,28,6,157,24,6,165,203,221,28,6,240,10,169,0,145,203,230,203,24
0
2030 DATA 42,208,239,189,16,6,170,189,48,6,133,205,189,49,6,133,206,177,205,240,14,145,203,230,205,208,2
2040 DATA 230,206,230,203,240,10,208,238,169,0,145,203,230,203,208,250,230,204,166,207,232,134,207,224,4,144,1
54
2042 DATA 173,15,208,240,34,106,144,2,162,0,106,144,2,162,1,106,144,2,162,2,169,0,157,28,6,157,32,6
2044 DATA 141,30,208,238,39,6,169,2,141,132,6
2046 DATA 173,7,208,240,3,141,38,6
2050 DATA 174,252,6,240,36,206,248,6,16,31,189,0,6,141,0,210,232,189,0,6,141,1,210
2052 DATA 232,189,0,6,240,9,141,248,6,232,142,252,6,208,3,141,252,6
2054 DATA 174,253,6,240,36,206,249,6,16,31,189,0,6,141,2,210,232,189,0,6,141,3,210
2056 DATA 232,189,0,6,240,9,141,249,6,232,142,253,6,208,3,141,253,6
2080 DATA 104,168,104,170,104,76,98,228
2090 GOSUB 3010:GOSUB 4100:RETURN
3000 REM * PAGE 6 - INSERT VBI ROUTINE *
3010 RESTORE 3020:FOR I=1536 TO 1545:READ A:POKE I,A:NEXT I
3020 DATA 104,160,0,162,0,169,7,76,92,228
3030 A=ADR(VB$):B=INT(A/256):C=A-256*B:POKE 1538,C:POKE 1540,B:RETURN
4099 REM ** SOUNDS **
4100 RESTORE 4120:FOR I=1665 TO 1766:READ A:POKE I,A:NEXT I
4120 DATA 60,170,3,54,170,3,48,170,3,45,170,3,40,170,3,36,170,3,0,0,0
4132 DATA 10,204,3,11,204,3,10,204,3,9,204,3,8,204,3,7,204,3,0,0,0
4142 DATA 20,143,1,80,143,3,90,140,6,95,137,20,100,134,30,0,0,0
4144 DATA 13,12,20,16,143,30,12,8,40,16,132,60,23,130,70,0,0,0
4146 DATA 40,174,20,50,172,20,60,170,20,70,168,20,80,166,20,90,164,20,105,162,20,0,0,0
4200 REM ** IMAGES **
4210 DIM V0$(4):RESTORE 4220:FOR I=1 TO 4:READ A:V0$(I)=CHR$(A):NEXT I
4215 V0=ADR(V0$):POKE IMGPT+1,INT(V0/256):POKE IMGPT,V0-256*PEEK(IMGPT+1)
4220 DATA 36,90,153,0
4230 DIM V1$(5):RESTORE 4240:FOR I=1 TO 5:READ A:V1$(I)=CHR$(A):NEXT I
4235 V1=ADR(V1$):POKE IMGPT+3,INT(V1/256):POKE IMGPT+2,V1-256*PEEK(IMGPT+3)
4240 DATA 66,165,24,24,0
4250 DIM V2$(5):RESTORE 4260:FOR I=1 TO 5:READ A:V2$(I)=CHR$(A):NEXT I
4255 V2=ADR(V2$):POKE IMGPT+5,INT(V2/256):POKE IMGPT+4,V2-256*PEEK(IMGPT+5)
4260 DATA 195,36,24,24,0
4270 DIM V3$(6):RESTORE 4280:FOR I=1 TO 6:READ A:V3$(I)=CHR$(A):NEXT I
4275 V3=ADR(V3$):POKE IMGPT+7,INT(V3/256):POKE IMGPT+6,V3-256*PEEK(IMGPT+7)
4280 DATA 129,66,36,24,24,0
4290 DIM V4$(6):RESTORE 4296:FOR I=1 TO 6:READ A:V4$(I)=CHR$(A):NEXT I
4292 V4=ADR(V4$):POKE IMGPT+9,INT(V4/256):POKE IMGPT+8,V4-256*PEEK(IMGPT+9)
4296 DATA 255,255,255,255,255,0
4300 RETURN
4999 REM ** CHANGE DISPLAY LIST **
5000 DL=PEEK(560)+256*PEEK(561)
5010 POKE DL+3,70:POKE DL+6,6:FOR I=DL+7 TO DL+28:POKE I,4:NEXT I:RETURN
6000 GRAPHICS 0:POKE 752,1:POSITION 8,1:? "***  VULTURES III  ***"
6010 POSITION 5,3:? "The vultures are landing and ":? "removing protective layers above"
6020 ? "you!  There are three layers and ":? "every time three vultures land":? "one layer will disappear.":?
6030 ? "   You can stop the birds with":? "a removal device controlled by":? "joystick 0. You get ten times"
6040 ? "the difficulty for each bird you":? "stop.  When you have removed a"
6050 ? "flock, the difficulty will go up.":? :? "If you get hit by lightning your"
6060 ? "score will decrease by a hundred":? "times the difficulty.":? :? "Just a sec..."
6090 RETURN
```

K-razy Antiks

Game Review

by Sheldon Leemon

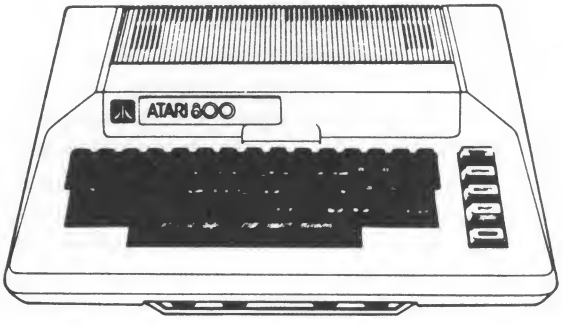
Don't be confused by the pun. The "ANTIC" that everyone associates with Atari computers is the support chip that makes possible the superb graphics needed for all of those neat arcade-type games. The "Antiks" in the title of this product refers to the insect you need in order to have a picnic. When the two get together, you wind up with a neat arcade-type game with great graphics, and everyone has a picnic!

Krazy Antiks is the fourth game-cartidge released for the Atari 400/800 computers, but it bucks the trend of "me-too" arcade-style games. Lately it seems that everyone is trying to cash in on the arcade craze by serving the warmed-over remains to computer owners. We have seen Bug Attack, Millipedes, Megalegs and Atari's own Centipede competing for the insect-eradication dollar. awbreaker, Pacman, and Ghost Hunter have been trying to satisfy the appetite of gamers hot to gobble. Why did the frog cross the highway and the river? Ask the makers of Pacific Coast Highway, Preppie, and now Frogger. Even K-Byte's earlier ventures into game programming tended to follow the heavily beaten path. But Antiks has just enough of a twist to be considered a new idea in a market saturated with retreads.

I must concede that the locale of the action is nothing novel--the ant hill in question strongly resembles the type of maze used in any number of games spawned by that prolific procreator, Pac-man. But the scenario is a fresh one. You play the role of the White Ant, and your purpose is one familiar to students of biology--to perpetuate the survival of the species. You start the game with thirty or so eggs at the bottom of the screen, which represent your capacity to reproduce. Arrayed against you are several adversaries. First, one ant each of the four basic ant types, the yellow, blue, green and red ant, circulate around the maze, trying to devour you. Another natural enemy is the dreaded anteater, who strolls into the picture every so often and inserts his long tongue into the anthill at random, sucking up friend foe, ant and egg alike. Finally, periodically a rain shower turns the lower part of the anthill into a miniature Wilkes-Barre, only without the federal aid.

With the odds against her, the lone ant has little chance for individual survival for long. Fortunately, if she can find a safe place in the maze in which to lay an egg where it will not be eaten by another ant, then after she is gone, the egg will hatch, and another white ant will take her place. Moreover, she has a weapon she can use. The other ants are busy laying eggs also, and when she eats one of theirs, she begins to glow, letting you know that the next egg she lays will be deadly to the other ants, if laid directly in their path. At each level, play continues until the white ant is killed, without leaving any eggs in the maze, or until all four of the other ants have been killed without surviving offspring. If the latter occurs, the game proceeds to the next level, and four new enemy ants come marching in to the tune of "When Johnny comes Marching Home" (which some like to think of as "The Ants Come Marching Two by Two"). When you succeed in clearing a level, don't become overconfident. Each maze has 99 levels of difficulty! If that fails to provide enough variation, there are a total of 6 different maze configurations to try out.

The play values are very good in Krazy Antics. Even an experienced player can get caught early on by a freak accident, which inevitably leads to "just one more" game. There is a pause option, for those disturbed by the unwarranted intrusions of friends and family. Unfortunately, there is no multi-player option. But if you don't mind going it along, you'll bless the day that ant met ANTIC.



(continued)

PUZZLE CORNER

by Charles Godfrey

RECAP OF PUZZLE #2

Shipwrecked in a hurricane, five saliors in a rowboat landed on a small island inhabited by one monkey and numerous coconut trees. The trees had just finished bearing a bumper crop of coconuts, all of which had blown to the ground in the hurricane.

Having nothing better to do, the sailors decided to gather all the coconuts on the island into a single pile. They finished the job by nightfall, and then fell soundly asleep. But one sailor awoke in the middle of the night and became worried that he wouldn't receive his fair share of the milky palm fruit. He stole away to the pile and removed exactly his fair share of the coconuts, buried them, and went back to sleep. Another sailor awoke a short time later and did the same--buried his fair share of the coconuts and returned to sleep. Each of the three remaining sailors, in turn, did likewise.

The next morning, not suspecting what each of the others had done, the sailors divided the remaining coconuts, and each received his equal share. One coconut remained, which they gave to the monkey.

The challenge was to write a program to determine the total number of coconuts in the original pile.

PUZZLE #2 SOLUTION

Coconuts, coconuts. The island was full of coconuts. Even the monkey got a coconut. There were no out of state entries this time, but those puzzle gurus from last month, Burt Gregory and Chris and Mary Ratkowski were at each other again with their entries only 1 day apart. Burt used a 1 line APL program again on his monster IBM mainframe to solve the puzzle in 1.14 seconds. Chris and Mary were the winners though, using a 5 line Atari basic program. Claus Buchholz of Muskegon thought the puzzle was too easy, and solved it with pencil and paper with an excellent explanation of the mathemathics involved. Several people, like Chris and Mary, did some paper work prior to programming, solving for a base number to make the final program more efficient. Their formula goes something like this:

```
remainder = 4/5(4/5(4/5(4/5(4/5coconuts))))
          = (4/5)^5 coconuts
          = 1024/3125 coconuts
```

The total coconuts must be an even multiple of 3125 since all divisions during the night came out equal. The lowest correct answer is 12500 coconuts. By brute force, the following program would slowly but surely crank out the correct answers:

```
100 X=0
110 Z=X
120 D=0
130 IF Z <> 5*INT(Z/5) THEN 190
140 Z=Z-Z/5
150 D=D+1
160 IF D<5 THEN 130
170 IF Z-5*INT(Z/5)<>1 THEN 190
180 PRINT X;" IS A SOLUTION"
190 X=X+1
200 GOTO 110
```

MACE PUZZLE #5

This month I ran out of imagination and am borrowing a puzzle from a book by Donald D. Spencer. Be prepared to let your computer run a LONG time for this one.

Once there was a king who thought of himself as quite a mathematician. He told a prisoner, "Give me a problem to solve, and you may go free until I solve it. But as soon as I have the answer, off comes your head!" Now the prisoner was rather clever himself and here is the problem he gave the king.

The numbers 220 and 284 are amicable numbers. The sum of the proper divisors of 220 equal 284

1+2+4+5+10+11+20+22+44+55+110 = 284

and the sum of the proper divisors of 284 equals 220. Find the next pair of amicable numbers.

The story goes that the prisoner went free and finally died of old age because the king never solved the problem.

What I am looking for in this puzzle is the longest list of amicable numbers submitted. The Catch: To prove you did not get the answers from some book, your entry must be a computer printout accompanied by the program.

(continued)

Send your answers to:

Charles Godfrey
29646 Chelmsford
Southfield, MI 48076
559-1272

AUTHORS DETERMINE WINNERS- The authors of the MACE Puzzles presented here are responsible for determining the winners to their respective puzzles. Any questions concerning a puzzle should be directed to the particular author involved. MACE will award the prizes either in person at a meeting, or by mail. Contestants must be sure to CLEARLY print your name, address, and telephone number so that we can contact you if necessary.

Anyone wishing to submit a puzzle may do so. Write or call the puzzle master for further information.

Forth Language SIG Report

by Todd A. Meitzner, Manager SIG/FORTH MACE

SIG/FORTH held its October meeting on October 6th. It was the first meeting to be held at the Royal Oak campus of the Oakland Community College. The meeting began shortly after 7:00 with general socializing and then continued with the release of Mesa-Forth to the members.

Some explanation of how to use it was also given. Then a talk was given on some of the various error codes found in Forth. The meeting eventually ended up in a general discussion of Forth and other subjects.

The December meeting will be held on Wednesday, December 1st. Note that the date in the sig report last month was incorrect. The meeting place will be at the Royal Oak campus of the Oakland Community College in Room B 119, the employee lounge. This room is located in the Administration section B off the main hallway on the first floor. There are two archways into this section and the room can be found by going through either to the last hallway. The room is at one of the corners of the last hallway.

As yet there is no meeting place for the January meeting but it is hoped we can once again obtain the use of a room at OCC.

The Sig/Forth meetings are open for all MACE members to attend. We hope to see you there. If anyone wishes to contact me for further information my phone number is listed under the MACE SIG GROUPS section elsewhere in this newsletter.

Assembler Language SIG
By Phil Heavin, Secretary
MACE SIGASM

October Meeting Minutes

Our business meeting was kept even shorter than usual this month to allow more time for the rest of the meeting. Tom welcomed several new people who were attending a SIGASM meeting for the first time. Next, Tom conducted a short session describing the application of the 6502's register manipulation instructions, load, store and transfer. In addition Tom described the basic structure for an assembly language program. In addition to being good programming practice, it was also a good method for overcomming 'terminal fright' when sitting down to begin writing a program.

Next we spent two hours discussing, in DETAIL an assembly language program that I wrote for SIGASM to demonstrate how to write an assembly language game program. This program demonstrated - a custom display list, smooth player motion done in a VBLANK routine under joystick control, player-playfield priority, and autorun on binary load. Luckly, without much planning this session demonstrated several of the points from Tom's sessions from the last two meetings.

December's Meeting

December's meeting will be Thursday, the 2nd at the home of Kent and Karen Kujala in Taylor. You can contact them at 292-9414 or me at 939-6213. The meeting will begin at 7:00 with socializing and free form discussion with the actual business portion starting at 7:30. We hope to see you there.

MICRO CHIP Your complete computer gaming store with over 500 exciting titles in discs and cartridges. At MICRO CHIP you can play the game on our systems before you buy it.

If we don't have . . . we'll order it with an additional 10% off our everyday low price. Yes, we are competitively priced with discount prices!

MICRO CHIP INC.

MICRO CHIP INC.
3628 ROCHESTER RD. TROY, MI
CENTURY PLAZA SHOPPING CENTER
PHONE: (313) 524-1230
HOURS: MON. – SAT. 10-9 SUN. 10-5

ld Of... MICRO CHIP

R ROCHESTER RD.

OY, MICHIGAN

E [313] 524-1230

CARTRIDGES

ACTIVISION
COLECO
INTELLIVISION®
ODYSSEY 2
SPECTRAVISION
IMAGIC
M NETWORK
PARKER BROTHERS
ATARI
U.S. Games

DISCS

ADVENTURE INTERNATIONAL
ASTAR INTERNATIONAL
AUTOMATED SIMULATIONS
AVALON HILL GAME COMPANY
AVANT-GARDE CREATIONS
BEAGLE BROTHERS
BRODERBUND SOFTWARE
BUDGE CO.
BUDGECO
CALIFORNIA PACIFIC
CAVALIER
CONTINENTAL SOFTWARE
C P U SOFTWARE
CPU SOFTWARE
DAKIN 5/LEVEL 10
DATAMOST
DELTA SOFTWARE
DON'T ASK SOFTWARE
DOUBLE GOLD SOFTWARE
EDU-WARE
GEBELLI SOFTWARE
HAYDEN SOFTWARE
HIGHLANDS COMPUTER SERVICES
INFOCOM
INNOVATIVE DESIGN SOFTWARE
THE LEARNING COMPANY
THE LOGICAL CHOICE
MICROLAB
MICROSOFT
MUSE
ON-LINE SYSTEMS
PICCADILLY
PHOENIX SOFTWARE
QUALITY SOFTWARE
RIVERRANK SOFTWARE
SENSIBLE SOFTWARE
SENTIENT SOFTWARE
SIERRA SOFTWARE
SIRIUS SOFTWARE
SOFTAPE
SOFTWARE EMPORIUM
SPINNAKER SOFTWARE
STONEWARE PRODUCTS
STRATEGIC SIMULATIONS
SUBLOGIC
SYNERGISTIC SOFTWARE
TURNKEY SOFTWARE
UNITED SOFTWARE OF AMERICA
VERSA COMPUTING
VISICORP
VOYAGER SOFTWARE
ZEITGEIST
ACORN SOFTWARE
ARCADE PLUS
ARTSCI
ATARI INCORPORATED %
APX
ARX
ATARI PROGRAM EXCHANGE %
COMPUTER MAGIC, LTD.
DATASOFT
IN-HOME SOFWARE
JV SOFTWARE
K-RYTE
PRISM COMPUTERS
PROGRAM DESIGN INC.
ROKLAN CORP.
SYNAPSE SOFTWARE
SYNCRO

"SPECIALS"

HOT ... HURRY, WHILE THEY LAST

	OUR$	LIST$
FROGGER BY ON-LINE	24.95	34.95
ATHANA BLANKS 5¼" SSDD	20.95	45.95

MICRO COUPON

ATARI 400/800, APPLE SOFTWARE ONLY

10% OFF

YOUR PURCHASE

(EXCEPT FOR ADVERTISED SPECIALS)

COUPON EXPIRES ON 11-30-82. LIMIT ONE PER PERSON.

MICRO COUPON

MACE SIG GROUPS

MACE offers members the opportunity to explore specific applications of Atari Computing in Special Interest Groups (SIGS) where MACE folks with common areas of interest can meet more informally than would be possible at our general membership meetings.

The following groups have registered as official MACE Special Interest Groups:

SIG/ASSEMBLER
Manager: Tom Hunt
Secretary: Phil Heavin
939-6213

SIG/BASIC
Manager: Jim Spitzer
543-0961

BUSINESS SYSTEMS
Manager: Douglas Perenchio
776-7626

SIG/EDUCATION
Manager: Mark Davids
774-9709

SIG/FORTH
Manager: Todd Meitzner
542-1752

SIG/GAMES
Manager: Stephen Tobias
979-5740

SIG/GRAPHICS
Manager: Ken Hein
254-1761

SIG/HARDWARE
Manager: Chris Ratkowski
532-5421

SIG/NEW USERS
Manager: Michael Winters
645-2193

UTILITIES
Manager: Charles Godfrey
559-1272 Home
362-9110 Work

Get involved!! Join a MACE SIG today!!!

An Advanced Use of the ATARI Macro Assembler

by Phil Heavin
MACE Assembler SIG

As promised, this article will cover a more advanced application of macros. I will assume that you have at least read my previous article on macros. It would be even better if you have read the manual "ATARI MACRO ASSEMBLER". The following example shows how macros can be used to implement something fairly complicated. It is NOT necessairly the best way to implement the desired commands, as a matter of fact several possible improvements are noted below.

THE EXAMPLE COMMANDS

The commands to be implemented are the following:

CALL SUBNAM[,ARG1,ARG2....]

Where SUBNAM is the name of a subroutine being called and ARG1 through ARG8 are optional arguments to be passed to the called subroutine.

Each argument can be one of the following forms:

labref - label reference, the address of 'labref' will be passed as the argument

[word] - the 16 bit value will be located in local storage by the CALL macro and its address as the argument.

[byte] - the 8 bit value will be stored as the first byte of two bytes located in local storage by the CALL macro and its address as the argument.

SUBR SUBNAM[,A][,X][,Y]

This command is the first command of a subroutine. The optional arguments, A, X, and Y specify which registers should be saved on entrance to the subroutine and restored when returning from the routine.

RETURN

This command specifies the logical end of the subroutine and will return control to the first executable statement after the CALL. Before control is actually returned to the calling program, any register that was saved by the SUBR command will be restored.

DESIRABLE FEATURES

1. Simple to use
2. A, X, Y should pass through
3. Reentrant (if a call is in process in the main program, the VBLANK routine which interrupts it should also be able to do a call)
4. Recursive, (A routine should be able to call itself with arguments)5. Calls should be nestable.
6. Should support null arguments and variable length argument lists.

THE IMPLEMENTATION CHOSEN

There are many ways to implement argument passing for subroutines. The main thing that I wanted to accomplish was to be able to pass the address of arrays of data and make it as simple as possible for the subroutine to access the data in the arrays.

A secondary requirement was to support lengthy argument lists called to several levels.

So, how well does the macro implementation meet the criteria listed above?

1. Simple to use?
   Yes, as outlined above.

2. Do A, X, Y pass through?
   Yes, they will be passed through the calls transparently.

3. Reentrant?
   Not quite, the local variable CATEMP can get

(continued)

clobbered under this condition.

4. Recursive?
Sort of. All of the subroutine call macros are, but it is up to you to manage your own local data storage to make the routine fully recursive.

5. Calls nestable?
Yes.

6. Null arguments and variable length argument lists?
Yes.

MEMORY REQUIREMENTS

Page zero -

1 byte for CATEMP
2 bytes for the address of the argument stack.

plus

2 bytes for each argument. Must be labeled ARG1...ARG8. They must be located in consecutive locations on page zero. You only need enough for the longest argument list you intend to use.

At the end of your program -

The argument stack which will grow from the end of your program toward higher addresses.

THE MACROS

In addition to those described above, there is an initialization macro which must be invoked once at the top of your program

```
CALLDF  STACKT,PZERO
```

Where:

STACKT - a label at the very end of your program for the start of the argument stack.

PZERO - address on page zero of 3 bytes for the macros to use.

Now the macros themselves.

```
;
                        ;CALLDF MACRO
;
CALLDF: MACRO STACKT,PZERO
CATEMP = %2
CASTAK = %2+1
        LDA #LOW[%1]
        STA CASTAK
        LDA #HIGH[%1]
        STA CASTAK+1

        ENDM
```

This macro generates the necessary equates and initializes the value of CASTAK the stack pointer.

```
;
;                       CALL MACRO
;
CALL:   MACRO   NAME,A1,A2,A3,A4,A5,A6,A7,A8
        STA     CATEMP
MAXARG  SET     0
        MULT    CHKMAX,%2,%3,%4,%5,%6,%7,%8,%9
        PUSHAG
        MULT    NEWARG,%2,%3,%4,%5,%6,%7,%8,%9
        DOJSR   %1,%2,%3,%4,%5,%6,%7,%8,%9
        POPARG
        ENDM
```

This is the top level macro defining the CALL statement. This is accomplished as follows:

1. Save the A register in CATEMP.
2. Zero MAXARG which will be used to count the number of arguments.
3. Invoke CHKMAX for each argument to count the number of arguments.
4. Invoke PUSHAG to push the current values in ARG1 through ARGn onto the argument stack. (n=MAXARG)
5. Invoke NEWARG for each new argument supplying a new value for ARG1 through ARGn.
6. Invoke DOJSR to simulate a JSR.
7. On return from the called subroutine, invoke POPARG to restore ARG1 through ARGn to their state before the CALL.

```
;
;                       MULT MACRO
;
```

(continued)

```
MULT    MACRO   MNAME,A1,A2,A3,A4,A5,A6,A7,A8
        %1      1,%2
        %1      2,%3
        %1      3,%4
        %1      4,%5
        %1      5,%6
        %1      6,%7
        %1      7,%8
        %1      8,%9
        ENDM
```

This macro is invoked, as shown above, and will invoke the named macro once for each argument.

```
;
;                       CHKMAX MACRO
;
CHKMAX  MACRO   N,ARG
 IF [LSTR['*%2']>1]
MAXARG  SET     %1
 ENDIF
        ENDM
```

This macro will be invoked via MULT from the CALL macro. For any argument that is not null, MAXARG will be updated. Therfore, after it has been invoked for all arguments, MAXARG will contain the number of arguments used in the CALL statement.

```
;
;                       PUSHAG MACRO
;
PUSHAG  MACRO   ANUM
 IF MAXARG>0
        TYA
        PHA
        LDY     #[MAXARG*2]
?MZK    LDA     ARG1-1,Y
        STA     (CASTAK),Y
        DEY
        BNE     ?MZK
        CLC
        LDA     CASTAK
        ADC     #[MAXARG*2]
        STA     CASTAK
        BCC     ?NZK
        INC     CASTAK+1
?NZK
        PLA
        TAY
 ENDIF
        ENDM
```

This macro moves all the current arguments to the argument stack and increments the stack pointer accordingly. This could be improved by generating inline code for shorter argument lists and this loop only on the larger number of arguments.

```
;
;                       NEWARG MACRO
;
NEWARG  MACRO   ARGNUM,ARGADD
 IF %1<=MAXARG
  IF LSTR['*%2']>1
        PUTARG  %1,%2
  ELSE
        LDA     #0
        STA     ARG%1
        STA     ARG%1+1
  ENDIF
 ENDIF
        ENDM
```

For each argument in the CALL statement that isn't null invoke PUTARG to put the correct value in the argument pointers on page zero. Null arguments will have zero placed on page zero for that argument.

```
;
;                       PUTARG MACRO
;
PUTARG  MACRO   ARGNUM,ARGADD
   IF ['[       '<'%2'] AND ['%2'<'[]]]]]]]]']
        LDA     #LOW[?AZK]
        STA     ARG%1
        LDA     #HIGH[?AZK]
        STA     ARG%1+1
AZ1@N   MACRO
?AZK    DW      %2
        ENDM
   ELSE
        LDA     #LOW[%2]
        STA     ARG%1
        LDA     #HIGH[%2]
        STA     ARG%1+1
   ENDIF
        ENDM
```

Normal arguments are handled in the ELSE clause by moving their address on to the pointer variables on page zero. Direct arguments

(continued)

(indicated by surrounding the argument with brackets [and]) will be stored in local storage after the simulated JSR and its address placed on the page zero pointer variables.

This macro contains an example of defining a macro within a macro. The macro named An@N (where n=arg #) will be invoked in DOJSR to define the local storage for a direct argument.

```
;
;                          DOJSR MACRO
;
DOJSR   MACRO   NAME,A1,A2,A3,A4,A5,A6,A7,A8
        LDA     #HIGH[?RZK-1]
        PHA
        LDA     #LOW[?RZK-1]
        PHA
        LDA     #MAXARG
        PHA
        LDA     CATEMP
        JMP     %1
        MULT    TEMARG,%2,%3,%4,%5,%6,%7,%8,%9
?RZK:
        ENDM
```

This macro pushes the return address minus one (read how JSR and RTS work) onto the stack then the number of arguments, restores the value of the A register and jumps to the subroutine.

MULT is used to invoke TEMARG to allocate local storage for any direct arguments.

```
;
;                          TEMARG MACRO
;
TEMARG  MACRO   ARGNUM,ARGADD
  IF LSTR['%%2']>1
   IF ['[       '<'%2'] AND ['%2'<'[]]]]]]]]']
       A%1@N
   ENDIF
  ENDIF
       ENDM
```

For all direct arguments generate the DW statement by invoking the macro defined in the PUTARG macro.

```
;
;                          POPARG MACRO
;
POPARG  MACRO
 IF MAXARG>0
        PHA
        TYA
        PHA
        SEC
        LDA     CASTAK
        SBC     #[MAXARG*2]
        STA     CASTAK
        BCS     ?NZK
        DEC     CASTAK+1
?NZK
        LDY     #[MAXARG*2]
?MZK    LDA     (CASTAK),Y
        STA     ARG1-1,Y
        DEY
        BNE     ?MZK
        PLA
        TAY
        PLA
 ENDIF
        ENDM
```

This macro restores the arguments from the argument stack and sets the stack pointer back to their state before the CALL statement.

AS in the PUSHARG macro this macro could be optimized for fewer number arguments.

```
;
;                          SUBR MACRO
;
SUBR: MACRO NAME,SAV1,SAV2,SAV3
SUSAVA SET ['%%2'='%A']OR['%%3'='%A']OR['%%4'='%A']
SUSAVX SET ['%%2'='%X']OR['%%3'='%X']OR['%%4'='%X']
SUSAVY SET ['%%2'='%Y']OR['%%3'='%Y']OR['%%4'='%Y']
        PROC
:NUMARG: DB 0
%1:
        SUBR1
        ENDM
```

This macro expands the SUBR statement. It first sets the variables indicating which registers were requested to be saved. Next the SUBR1 macro is invoked. This rather artifical split is to accomodate the 255 character limit mentioned in my first article.

(continued)

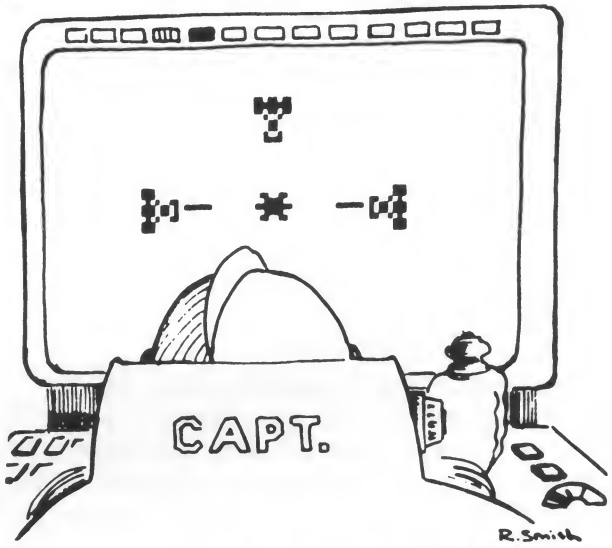
IS YOUR ATARI GETTING BORED?

EVEN A COMPUTER NEEDS A VACATION! TRY TAKING IT TO PRETZELLAND!

THE FIRST AMUSEMENT-THEME PARK FOR COMPUTERS ONLY!

SPACE WORLD—

STARBASE ASSAULT



BLAST THE ALIENS-
BEFORE THEY BLAST YOU!

VISIT-

ACTION-ADVENTURE WORLD—

ANDROID ATTACK

SHOOT YOUR WAY DOWN TO SAVE THE RUNAWAY NUCLEAR REACTOR-
THEN SAVE YOURSELF!

FANTASY WORLD—

AAARRRGGG!!!

CATCH 'EM IF YOU CAN-
BUT DON'T GET POISONED!

TAKE HOME A SOUVENIR OF YOUR VISIT-
FROM YOUR LOCAL DEALER, OR ORDER
DIRECT FROM PRETZELLAND, ONLY $18.95
SPECIFY 16K CASSETTE, OR 32K DISK

PRETZELLAND SOFTWARE

2005 E WHITTAKER RD.
YPSILANTI, MI 48197
(313) 483-7358

```
;
;                    SUBR1 MACRO
;
SUBR1: MACRO
       STA CATEMP
       PLA
       STA :NUMARG
 IF SUSAVA
       LDA CATEMP
       PHA
 ENDIF
 IF SUSAVX
       TXA
       PHA
 ENDIF
 IF SUSAVY
       TYA
       PHA
 ENDIF
       LDA CATEMP
       ENDM
```

This macro saves the A register pulls the number of arguments from the hardware stack and puts it into the local variable :NUMARG. It then saves any of the requested arguments then restores the A register to its original value.

```
;
;                    RETURN MACRO
;
RETURN: MACRO
 IF [SUSAVA=0] AND [[SUSAVX]OR[SUSAVY]]
       STA CATEMP
 ENDIF
 IF SUSAVY
       PLA
       TAY
 ENDIF
 IF SUSAVX
       PLA
       TAX
 ENDIF
 IF [SUSAVA=0] AND [[SUSAVX]OR[SUSAVY]]
       LDA CATEMP
 ENDIF
 IF SUSAVA
       PLA
 ENDIF
       RTS
       EPROC
       ENDM
```

If A register restoration was not requested and the current contents are going to be clobbered save it in CATEMP. Restore X or Y if restoration was requested. Next, put back either the current value of A or the value of A on entrance to the subroutine and return.

EXAMPLES OF EXPANDED CODE

The following examples show the code produced by a couple of different invocations of these macros. It should be noted that these examples are not of actual executable program, so don't try to figure out the purpose of the subroutines.

This is storage allocation and initialization that would appear at the top of the program.

```
 LIST G,-M

 ORG $C0
ARG1    DW      0
ARG2    DW      0
ARG3    DW      0
ARG4    DW      0
ARG5    DW      0
ARG6    DW      0
ARG7    DW      0
ARG8    DW      0

 ORG $1F00

 CALLDF PRGEND,$B0
CATEMP = $B0
CASTAK = $B0+1
       LDA #LOW[PRGEND]
       STA CASTAK
       LDA #HIGH[PRGEND]
       STA CASTAK+1
```

This call statement:

```
     CALL SUBY
```

Will produce this code:

```
       STA     CATEMP
       LDA     #HIGH[?R005A-1]
       PHA
       LDA     #LOW[?R005A-1]
       PHA
       LDA     #MAXARG
```

(continued)

```
        PHA
        LDA     CATEMP
        JMP     SUBY
?R005A:
```

This call statement:

```
CALL SUBAX,AL,[$5E],BL
```

Will produce this code:

```
        STA     CATEMP
        TYA
        PHA
        LDY     #[MAXARG*2]
?M002D  LDA     ARG1-1,Y
        STA     (CASTAK),Y
        DEY
        BNE     ?M002D
        CLC
        LDA     CASTAK
        ADC     #[MAXARG*2]
        STA     CASTAK
        BCC     ?N002D
        INC     CASTAK+1
?N002D  PLA
        TAY
        LDA     #LOW[AL]
        STA     ARG1
        LDA     #HIGH[AL]
        STA     ARG1+1
        LDA     #LOW[?A0032]
        STA     ARG2
        LDA     #HIGH[?A0032]
        STA     ARG2+1
        LDA     #LOW[BL]
        STA     ARG3
        LDA     #HIGH[BL]
        STA     ARG3+1
        LDA     #HIGH[?R003A-1]
        PHA
        LDA     #LOW[?R003A-1]
        PHA
        LDA     #MAXARG
        PHA
        LDA     CATEMP
        JMP     SUBAX
?A0032  DW      [$5E]
?R003A  PHA
        TYA
        PHA
        SEC
        LDA     CASTAK
        SBC     #[MAXARG*2]
        STA     CASTAK
        BCS     ?N0045
        DEC     CASTAK+1
?N0045  LDY     #[MAXARG*2]
?M0045  LDA     (CASTAK),Y
        STA     ARG1-1,Y
        DEY
        BNE     ?M0045
        PLA
        TAY
        PLA
```

This SUBR statement:

```
SUBR SUBXY,X,Y
```

Will produce the following code:

```
        PROC
:NUMARG: DB 0
SUBXY:
        STA CATEMP
        PLA
        STA :NUMARG
        TXA
        PHA
        TYA
        PHA
        LDA CATEMP
```

Your subroutine code will go here. To index into a byte array that is the third argument, just load an index into the Y register and use indirect indexed addressing like this:

```
LDY    #4
LDA    (ARG3),Y
```

The above code would load the 5th byte of the array passed as ARG3.

Now this statement:

```
RETURN
```

Will generate this code:

```
STA CATEMP
PLA
TAY
PLA
TAX
LDA CATEMP
RTS
AL: DS 20
BL: DS 30
PRGEND:
    END
```

ATARI MAGAZINE ARTICLES - PART II

COMPILED BY
EDWARD MIDDLEBROOK

TITLE	AUTHOR	SUBJECT	MAGAZINE	ISSUE
CHICKEN - A GREAT GAME	ANTIC	PROGRAM - ARCADE	ANTIC	APR 82
INVADERS FROM OUTER SPACE	MARKLEY	PROGRAM - ARCADE	COMPUTE	APR 81
MAZE RIDER	BACHAND	PROGRAM - ARCADE	ANALOG	JAN 81
MIDNIGHT STRIP	CLAYTON	PROGRAM - ARCADE	MACE	JUL 82
MIDNIGHT STRIP	CLAYTON	PROGRAM - ARCADE	MACE	JUL 82
SHOOT	PALEVICH	PROGRAM - ARCADE	COMPUTE	SEP 81
SPACE INVADERS	LEEMON	PROGRAM - ARCADE	MACE	OCT 81
TYPING SHOOT	COMPUTE	PROGRAM - ARCADE	COMPUTE	NOV 81
ATARI SOUNDS TUTORIAL	WHITE	PROGRAM - DEMO	COMPUTE	JAN 82
BILLIARD BOUNCE	LAWS	PROGRAM - DEMO	COMPUTE	DEC 81
COLOR WHEEL FOR THE ATARI	BLACKA	PROGRAM - DEMO	COMPUTE	JUL 80
PUT A RAINBOW IN ATARI	TEDSEN	PROGRAM - DEMO	COMPUTE	FEB 82
STRING ART	MAIMAN	PROGRAM - DEMO	COMPUTE	DEC 81
THE FLUID BRUSH	BAKER	PROGRAM - DEMO	COMPUTE	JAN 81
TICKER TAPE MESSAGES	MARTELL	PROGRAM - DEMO	COMPUTE	FEB 81
TIMES SQUARE ON ATARI	HARRIS	PROGRAM - DEMO	COMPUTE	NOV 80
ULTRACUBE: SUPERCUBE REV	BENSON	PROGRAM - DEMO	COMPUTE	APR 82
CONCENTRATION	BRANNON	PROGRAM - GAME	COMPUTE	MAR 82
CRYPTOGRAM	MARCUSE	PROGRAM - GAME	COMPUTE	JAN 82
WORD HUNT	BAKER	PROGRAM - GAME	COMPUTE	MAR 82
WORD SEARCH	JONES	PROGRAM - GAME	COMPUTE	JAN 82
MAYPOLE	MARKOYA	PROGRAM - GRAPHICS	COMPUTE	NOV 81
MOIRE MAGIC	CERRUTI	PROGRAM - GRAPHICS	COMPUTE	FEB 82
SUPERCUBE UPDATE	KINNAMON	PROGRAM - GRAPHICS	COMPUTE	AUG 81
SUPERCUBE	STEINBERG	PROGRAM - GRAPHICS	COMPUTE	APR 81
BAKER STREET BYTES	GIZYNSKI	PROGRAM - MISC	MACE	JUL 82
BALANCE YOUR CHECKBOOK	BACHAND	PROGRAM - MISC	ANALOG	MAR 81
COMPUTER GREETING CARDS	VICTOR	PROGRAM - MISC	COMPUTE	JUN 81
MATCH GAME	WALKER	PROGRAM - MISC	COMPUTE	OCT 81
MONTHLY BAR GRAPH PROGRAM	WHITE	PROGRAM - MISC	COMPUTE	NOV 80
POEM WRITER	ROBERTS	PROGRAM - MISC	COMPUTE	AUG 81
PRESCHOOL LETTER PRACTICE	DUBIN	PROGRAM - MISC	MACE	MAY 82
TOWERS OF HANOI	KNOPMAN	PROGRAM - MISC	ANALOG	MAY 81
AN ATARI DISASSEMBLER	FORTNER	PROGRAM - UTILITY	COMPUTE	JUN 81
ATARI BASIC STRING SORT	WHITE	PROGRAM - UTILITY	COMPUTE	SEP 81
ATARI DISK FILE DUMP	BAKER	PROGRAM - UTILITY	COMPUTE	OCT 81
ATARI DISK MENU	LINDSAY	PROGRAM - UTILITY	COMPUTE	JAN 81
ATARI PROGRAM LIBRARY	MARCUSE	PROGRAM - UTILITY	COMPUTE	OCT 81
BAKER STREET BYTES	GIZYNSKI	PROGRAM - UTILITY	MACE	FEB 82
BAKER STREET BYTES	GIZYNSKI	PROGRAM - UTILITY	MACE	JAN 82
BINARY/DECIMAL CONVERSION	WHITE	PROGRAM - UTILITY	COMPUTE	JUN 81
BOOTABLE CASSETTE TO DISK	SCHULTZ	PROGRAM - UTILITY	MACE	NOV 81
COPY ATARI TO PRINTER	STRAW	PROGRAM - UTILITY	COMPUTE	JUN 82
COPY SCREEN TO PRINTER	STRAW	PROGRAM - UTILITY	COMPUTE	MAY 81
DATA MANAGEMENT SYSTEM	MARCUSE	PROGRAM - UTILITY	COMPUTE	NOV 81
DISK DIRECTORY PRINTER	LINDSAY	PROGRAM - UTILITY	COMPUTE	MAY 81
DISKDUP	SCHULTZ	PROGRAM - UTILITY	MACE	DEC 81
ERROR REPORTING SYSTEM	LINDSAY	PROGRAM - UTILITY	COMPUTE	NOV 80
LINE RENUMBERING UTILITY	GROPPER	PROGRAM - UTILITY	COMPUTE	MAR 81
LISTER	HARTMAN	PROGRAM - UTILITY	ANALOG	#4

ATARI MAGAZINE ARTICLES (continued)

TITLE	AUTHOR	SUBJECT	MAGAZINE	ISSUE
REVISION B ROM TEST	ROSER	PROGRAM - UTILITY	MACE	AUG 82
SCREEN SAVE ROUTINE	TREM	PROGRAM - UTILITY	COMPUTE	MAR 82
SUBROUTINE TO AID DEBUG	GREEN	PROGRAM - UTILITY	COMPUTE	JUN 82
SUPERFONT	BRANNON	PROGRAM - UTILITY	COMPUTE	JAN 82
SYS/STAT PROGRAM	HARTMAN	PROGRAM - UTILITY	ANALOG	MAY 81
SYSTEM CLOCK FOR ATARI	ZIMMERMAN	PROGRAM - UTILITY	COMPUTE	AUG 82
TERMINAL EMULATOR	BUTTERWORTH	PROGRAM - UTILITY	MACE	JUN 81
TEXTPLOT	BRANNON	PROGRAM - UTILITY	COMPUTE	NOV 81
VARIABLE NAME UTILITY	McGRAW	PROGRAM - UTILITY	COMPUTE	OCT 81
3-D GRAPHS FAST AND EASY	HUDSON	PROGRAM TECHNIQUES	ANALOG	#5
ACCURATE TIMING IN BASIC	NAVAS	PROGRAM TECHNIQUES	COMPUTE	APR 82
ATARI DATA	BAKER	PROGRAM TECHNIQUES	COMPUTE	JUL 81
ATARI TIMING DELAYS	CLARK	PROGRAM TECHNIQUES	COMPUTE	NOV 81
ATARI TUTORIAL: PARM 10	CRAWFORD	PROGRAM TECHNIQUES	BYTE	JUN 82
ATARI TUTORIAL: PART 5	CRAWFORD	PROGRAM TECHNIQUES	BYTE	JAN 82
BLINKING CHARACTERS	JONES	PROGRAM TECHNIQUES	COMPUTE	DEC 81
CARD GAMES IN GR. 1 & 2	SEIVERT	PROGRAM TECHNIQUES	COMPUTE	NOV 80
CHOOSE YOUR JOYSTICK	LINDSAY	PROGRAM TECHNIQUES	COMPUTE	JUL 80
CLEARING MEMORY	BRANNON	PROGRAM TECHNIQUES	COMPUTE	FEB 82
COLOR & LOCATE FOR PONG	GREENSPAN	PROGRAM TECHNIQUES	COMPUTE	SEP 81
COLOR & SOUND WITH PADDLE	SCHRIEBMAN	PROGRAM TECHNIQUES	COMPUTE	FEB 81
CONDENSING DATA STATEMENT	PATCHETT	PROGRAM TECHNIQUES	COMPUTE	MAY 81
DOWN MEMORY LANE	LEEMON	PROGRAM TECHNIQUES	MACE	DEC 81
DOWN MEMORY LANE	LEEMON	PROGRAM TECHNIQUES	MACE	AUG 81
DOWN MEMORY LANE	LEEMON	PROGRAM TECHNIQUES	MACE	FEB 82
EASY READING OF JOYSTICK	McMAHON	PROGRAM TECHNIQUES	COMPUTE	AUG 81
FORMATTED OUTPUT IN BASIC	WROBEL	PROGRAM TECHNIQUES	COMPUTE	MAR 81
FORMATTING INPUT	FREESE	PROGRAM TECHNIQUES	COMPUTE	NOV 81
GAME PROGRAMMING	OCKERS	PROGRAM TECHNIQUES	ANTIC	JUN 82
GRAPH IT ON THE ATARI	NEIL	PROGRAM TECHNIQUES	COMPUTE	OCT 81
GRAPHICS OF POLAR FUNCTNS	VELUDO	PROGRAM TECHNIQUES	COMPUTE	SEP 80
MAKE ATARI A BIT WISER	BANNON	PROGRAM TECHNIQUES	COMPUTE	MAY 81
MEMORY PROTECTION	CLARK	PROGRAM TECHNIQUES	COMPUTE	JUL 81
PRINT NUMBERS MAKES CENTS	HALCOMB	PROGRAM TECHNIQUES	COMPUTE	NOV 81
PROGRAM DEVELOPMENT	HOFFMAN	PROGRAM TECHNIQUES	COMPUTE	JUL 81
READ KEYBOARD ON THE FLY	BRUUN	PROGRAM TECHNIQUES	COMPUTE	SEP 80
RESTORING DATA ON ATARI	FRUMKER	PROGRAM TECHNIQUES	COMPUTE	AUG 81
STRING ARRAYS IN BASIC	BRANNON	PROGRAM TECHNIQUES	COMPUTE	APR 81
USE STRINGS FOR GRAPHICS	BOOM	PROGRAM TECHNIQUES	COMPUTE	MAY 81
USING CONSOLE SWITCHES	BRUUN	PROGRAM TECHNIQUES	COMPUTE	JAN 81
WHAT TO DO IF NO JOYSTICK	SCHULMAN	PROGRAM TECHNIQUES	COMPUTE	SEP 80
YOU'RE WASTING ARRAYS	BACHAND	PROGRAM TECHNIQUES	ANALOG	MAR 81
ATARI TUTORIAL: PART 7	FRASER	SOUND	BYTE	MAR 82
BASENOTES IN BASIC	WHITE	SOUND	ANALOG	MAY 81
MAKE MUSIC WITH ATARI	COLSHER	SOUND	KILOBAUD	JUN 82
ADDING VOICE TO PROGRAMS	VICTOR	TAPES	COMPUTE	JUL 80
ADDING VOICE TRACK PART 2	DOLEMAN	TAPES	COMPUTE	JUL 81
BOOT-TAPE GENER FROM DOS	POLONE	TAPES	COMPUTE	OCT 81
HELP FOR CASSETTE OWNERS	PHILLIPS	TAPES	ANTIC	APR 82
MACE TUTORIAL	GIESE	TAPES	MACE	APR 82

End of Part II

MICHIGAN ATARI COMPUTER ENTHUSIASTS

P.O. Box 2785
Southfield, MI 48037

24HR AMIS BULLETIN BOARD 589-0996

MACE HOTLINE (Meeting Info) 338-6837

PRESIDENT
Marshal Dubin
2639 Hempstead
Auburn Heights, MI 48057
338-6837

VICE-PRESIDENT
Jerry Aamodt
4148 Huhn
Rochester, MI 48063
574-1020

TREASURER
James Phillips
40008 Cambridge
Bldg 23 - Apt 103
Canton Township, MI 48187
981-1523

CORRESPONDING SECRETARY
Mike Lechkun
32229 Ruehle
Warren, MI 48093
978-8432

RECORDING SECRETARY
William Black
1251 Duckwood
Milford, MI 48042
887-6870

PROGRAM COORDINATOR
Gretchen Levitan
12709 Borgman
Huntington Woods, MI 48070
399-6964

DISK LIBRARIAN
Chet Gonterman
35088 Savannah Lane
Farmington Hills, MI 48018
553-7443

CASSETTE LIBRARIAN
Faris Ajo
6642 Leytonstone
West Bloomfield, MI 48033
661-0388

NEWSLETTER EDITOR
Arlan R. Levitan
12709 Borgman
Huntington Woods, MI 48070
Compuserve: 70675,463
Source ID: TCT987
Voice: 399-6964

8
7

NEXT MEETING 12/21/82

Southfield Pavillion
Ten & a Half & Evergreen
7:00 PM

M.A.C.E.
P.O Box 2785
Southfield, MI 48037

$15.00 fee for 12 months*
$20.00 after 12/31/82
Renew NOW and SAVE!

M.A.C.E. MEMBERSHIP APPLICATION

Name _____ Phone _____
Street _____
City _____ State _____ Zip Code _____
Company (if applicable) _____
System Description _____ Disk/Tape _____
Suggestions _____
I can help with . . . _____

[ ] NEW [ ] RENEWAL

If Renewal, MACE #_____

*Make checks payable to:
M.A.C.E.

Date _____ Coupon _____
Amount _____ Membership Card Number _____
☐ Cash Expires _____
☐ Check Number

Announcing the best Error Free Personal Computer Diskette Money can Buy. For Less.

- **Error Free**
- 1 year warranty
- Hub ring installed
- Write/Protect notch
- Next day delivery

$19.90/box of 10

- **No** minimum order quantity
- 8" or 5¼"
- Plus 2 Free BONUS DISKS/Box*

*5¼" SPECIAL ONLY

If you are a member of a user group or a school district please call for special terms on future offers. TSS is the largest specialty supplier of magnetic media in the Midwest. We have the products that you want when you need them. Please take advantage of this introductory offer and call us **now.**

TSS Transaction Storage Systems, Inc.
MAGNETIC MEDIA SPECIALISTS

CALL TOLL FREE
1-800-521-5700
1-800-482-4770 (Michigan)
313-557-3036 (Detroit)
312-922-0076 (Chicago)
614-221-1788 (Columbus)
513-621-1518 (Cincinnati)
Telex 810-224-4646

EXPECT A MIRACLE

YES, TSS is the magnetic media supplier that I have always wanted.

Please send me _____ *Box Quantity*, 8"☐ 5¼"☐, @ _____ *Price* *ea.*

I am interested. Please send me more information or call me at ( ) _____

For faster order entry call any of our toll-free or local numbers

Company _____

Name _____ *Title* _____

Address _____

City _____ *State* _____ *Zip* _____

Amex/Master Card/Visa orders are accepted *Expiration Date*

____ ______ ________ *month* _____ *year* _____

MICHIGAN ATARI COMPUTER ENTHUSIASTS
P.O. BOX 2785
SOUTHFIELD, MICHIGAN 48037

BULK RATE
U.S. POSTAGE
PAID
PERMIT #431
SOUTHFIELD, MI

01/83

IMPORTANT DATED MATERIAL

PLEASE DO NOT DELAY

Printing and Bindery Services by • GRAPHIC ENTERPRISES, INC. Detroit, Michigan • 313-839-6800